Secure Authentication System for Public WLAN Roaming

Yasuhiko Matsunaga[†]
Computer Science Division
Univ. of California, Berkeley
Berkeley, CA, U.S.A.

Ana Sanz Merino[†]
Computer Science Division
Univ. of California, Berkeley
Berkeley, CA, U.S.A.

Takashi Suzuki[‡]
Multimedia Laboratories
NTT DoCoMo, Inc.
Yokosuka, Kanagawa,Japan

Randy H. Katz[†]
Computer Science Division
Univ. of California, Berkeley
Berkeley, CA, U.S.A.

[†]{yasuhiko, asanz, randy}@eecs.berkeley.edu [‡]suzuki@spg.yrp.nttdocomo.co.jp

ABSTRACT

A serious impediment for seamless roaming between independent wireless LANs (WLANs) is how best to confederate the various WLAN service providers, each having different trust relationships with individuals and each supporting their own authentication schemes which may vary from one provider to the next. We have designed and implemented a comprehensive single sign-on (SSO) authentication architecture that confederates WLAN service providers through trusted identity providers. Users select the appropriate SSO authentication scheme from the authentication capabilities announced by the WLAN service provider, and can block the exposure of their privacy information while roaming. In addition, we have developed a compound layer 2 and Web authentication scheme that ensures cryptographically protected access while preserving pre-existing public WLAN payment models. Our experimental results, obtained from our prototype system, show the total authentication delay are well within 2 seconds. This is dominated primarily by our use of industry-standard XML-based protocols, yet are still small enough for practical use.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks - General**]: Data communications, Security and protection.

General Terms

Security, Design, Experimentation

Keywords

wireless LAN, hotspot, roaming, authentication, single sign-on, policy control, link layer security.

1. INTRODUCTION

Low deployment costs and high demand for wireless access have led to rapid deployments of public WLAN hotspot services by many providers, including startups and telecom operators [1]. Most service providers cannot cost-effectively deploy as many access points as needed to achieve good wide-area coverage, and thus supporting inter-operator roaming is a natural strategy for enlarging their service area. In such a roaming model, users may connect to the Internet via access points owned by providers that are unknown to them, for whom a trust relationship may not exist. Security mechanisms that protect both the user and the network are required. The roaming architecture works well in the cellular phone network because of its standard methods for determining user and service provider identity, and service accounting and settlement. Such a standardized architecture for authorization, access, and accounting, agreed to by a larger and more heterogeneous set of service providers, simply does not exist at this time for public WLANs. Even with mobility extensions to the Internet's routing protocols, the lack of such an architecture makes it difficult to federate WLAN service providers, leaving the user with the considerable burden of maintaining multiple identities and credentials.

Security always involves a tradeoff between convenience and risk. For public WLANs, users require that authentication, authorization and charging information (such as user unique identifier and credit card information) be protected against imprudent exposure to providers unless explicitly permitted by the user. At the same time the user may want seamless roaming by avoiding manual sign-on if it does not violate the user's security policy. From the provider's viewpoint, strict network access control is necessary to prevent theft of services from malicious attackers. On the other hand, WLAN providers normally give IP-level access to users before authentication to allow various authentication and authorization options, such as one-time credit card payment or to provide free local and advertisement content for non-subscribers. However, giving IP-level access without authentication yields a vulnerability to theft of service through IP or MAC address spoofing.

In light of these problems, we have developed a comprehensive security solution for public WLAN services, as an overlay on existing standard authentication and authorization models. We assume multiple underlying authentication methods;

we do not require a single method that is universally adopted. Our solution is designed to achieve three goals: 1) to confederate wireless LAN service provider under different inter-provider and user-provider trust relationships, 2) to protect user authentication information from unwilling exposure while also minimizing the amount of user intervention during sign-on, and 3) to strictly control network access by cryptographic methods while supporting the existing alternative authorization methods currently used in deployed public WLANs.

To achieve the first goal, we use single sign-on (SSO) authentication technologies to confederate WLAN service providers via trusted identity providers. As depicted in Fig. 1, we assume that there is at least one and possibly several identity providers with which the WLAN service providers have roaming agreements, and with which the user has strong trust relationships. Roaming agreements between a service provider and an identity provider typically include supported authentication protocols, configuration of secure communication channel, and the method of charging/revenue settlements. Examples of such identity providers are ISPs, credit-card companies, roaming service providers (wireless LAN aggregators), cellular network operators, etc. Some identity providers are already doing business in the public wireless LAN roaming market. The presence of such identity providers exempts users from having multiple identities and credentials, and helps users to roam across WLAN service providers under different levels of trust. The user is really authenticated by the identity provider, and the service provider, based on its trust relationship with the identity provider, relies on the authentication result. In Fig.1, the user can roam between the networks of WLAN service provider A and B because he has an account in identity provider 1, and both service providers have roaming agreements with identity provider 1. To roam into WLAN service provider C's network, the user should have an account at identity provider 2 because no other identity provider is available. Although the identity provider and the WLAN service provider are logically different entities, both of them may belong to a single administrative domain.

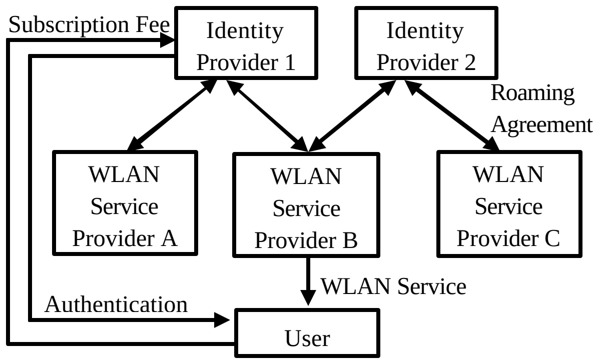

Fig. 1: Roaming Model

Our focus is to develop a decentralized authentication framework for inter-service provider roaming where each provider adopts its own authentication methods, and arbitrary roaming agreement relationships are made between identity providers and WLAN service providers. Our architecture is independent of the kind of authentication methods supported by identity or service providers, and allows users to choose the most preferred identity provider and authentication scheme. In particular, we considered two different SSO authentication standards in our architecture: RADIUS [2] and Liberty Architecture [3]. These authentication schemes were selected as examples because they represent the span of possible methods, commonly encountered (RADIUS) and other new schemes accommodated as they become available (Liberty). While RADIUS-based roaming is currently being deployed in public WLANs [4], Liberty-based roaming has an advantage of hiding the user's identity and credentials from weakly-trusted WLAN service providers. It should be pointed out that our approach is not limited to these authentication standards, but rather we have chosen multiple standards to insure that the architecture is flexible enough to handle heterogeneous underlying authentication methods.

For the second goal, we create an authentication flow adaptation framework in which the authentication scheme is chosen based on the user's authentication policy combined with the capabilities of underlying authentication server. A client-side policy engine determines the user authentication information based on user-defined policies (written in XML) and the communication context. The policy engine provides a generic API for authentication information access, which can be invoked not only by web-based mechanisms, but also by others such as link-layer authentication. In addition, the policy engine can support various access control models, such as requiring additional actions to be taken before authorization (a.k.a. provisional action) [5][6].

As for the third goal, we have developed a compound layer-2 (L2) and Web authentication scheme to ensure cryptographically protected access in public wireless LANs. In our compound scheme, the user first establishes a L2 session key by using a guest (anonymous) account in an IEEE 802.1X authentication. The user then embeds the L2 session key digest in web authentication. By binding the L2 and Web authentication results, our scheme prevents theft of service, eavesdropping, and message alteration in public WLANs.

To verify and evaluate our proposed architecture, we developed a federated WLAN testbed. Measurement results show that the additional delay by authentication adaptation process and the compound L2 and web authentication process are 320 msec and 120 msec, respectively.

The rest of this paper is organized as follows. Section 2 relates our design to previous works. Sections 3, 4, and 5 details single sign-on authentication, the design of authentication adaptation system, and the compound authentication scheme. Section 6 describes the prototype system and evaluation results. Finally, section 7 concludes the paper.

2. RELATED WORK

Link layer authentication

IEEE 802.1X standard [7] provides port-based access control based on the authentication results for devices interconnected by IEEE 802 LANs, and is considered a promising solution for securing corporate 802.11 networks. IEEE 802.11i [8] is being standardized to provide robust security in 802.11 wireless LANs, and its authentication scheme is based on 802.1X. 802.1X employs Extensible Authentication Protocol (EAP), on which any authentication mechanism can be used for authenticating both the user and the network and establishing L2 session key dynamically. Per-user encryption and message integrity check keys are derived from the L2 session key. Those keys protect the data packets sent over the air. Examples of authentication methods in the wireless LAN environment include EAP-TLS [9] and EAP-TTLS [10]. EAP-TLS uses client and server certificates for mutual authentication. EAP-TTLS also use certificate-based authentication for server authentication, and allows the user to use various client authentication schemes (e.g., MS-CHAPv2 [19]). Although these authentication methods work well in corporate WLAN environments, they exclude one-time credit-card authorization options and free advertisements in public WLANs, because they assume a pre-shared secret between user and network.

Web-based authentication and network layer access control

Many public WLAN providers employ web-based authentication in conjunction with IP packet filtering at a network access server based on the MAC and IP addresses. Since address spoofing is easily accomplished using readily available tools, this method does not provide strong security for network providers. Malicious users can monitor the wireless channel, acquire MAC and IP addresses of authenticated users, and send packets with spoofed addresses to perform theft of service or DoS attacks. Such an attack can be prevented by deploying IPsec Authentication Headers [11] to check for packet integrity and to control access based on its result, but at a substantial cost in bandwidth and computational overhead. To roam across different WLAN service providers, the authentication web server acts as a RADIUS client and forwards authentication messages to a RADIUS server in the user's home provider [4]. Although RADIUS-based roaming is being increasingly deployed, the user must show its identity and credentials to the WLAN service provider regardless of the level of trust relationship, due to HTTPS-RADIUS protocol conversion process at the service provider. Authentication schemes based on Zero knowledge proofs [12] or Secure Remote Passwords [13] can help hide user credentials at service providers by using a one-way hash function and ephemeral public keys. However, they do not help protect the user identifier from the service provider.

Several WLAN providers deploy their own proprietary network access client, not for security reasons but rather for user convenience to select the closest access point. They use their own authentication protocols in the serving network, and the authentication message flow is fixed regardless of user preference.

Finally, the CHOICE network [14] makes use of Microsoft Passport as a web authentication database, and uses a proprietary security sublayer between the link layer and IP layer. In their scheme, as a result of web authentication, the network gives the user terminal a (key, token) pair for the purpose of network access control thereafter. Our work differs from theirs in that an individual user can choose its own identity provider and use only standard-based link layer security, while they assume a centralized authentication server and require proprietary security sublayer. Moreover, our scheme makes it possible for a user to select the most preferred authentication method and identity provider among others, and prevent unwilling security information exposure according to the user's policy, which are not mentioned in their paper. This is mainly because our policy management function resides both in the network and the client, while their model assumed a policy controller only at the network side.

To summarize, existing web-based authentication does not provide sufficient level of security against the theft of service, dynamic selection of different authentication schemes, and user's policy-based protection of unwilling exposure of privacy information while roaming.

3. SINGLE SIGN-ON FOR CONFEDERATING WIRELESS LAN SERVICE PROVIDERS

In public WLANs, users are required to authenticate every time they roam into a different service provider's network. We make the following assumptions in designing our authentication architecture.

- There exists at least one certificate authority trusted by users, service providers, and identity providers.
- The user terminal can validate the certificate of the service provider's and identity provider's authentication servers.
- There are static trust relationships between the user and the identity provider, and between the service provider and the identity provider.
- The user can authenticate the service provider's authentication server via the identity provider's authentication server, and vice versa.

These assumptions are necessary for clients and service providers to authenticate each other to establish dynamic trust relationships.

Single sign-on (SSO) is a method for enabling users to access multiple systems after being authenticated just once, thus reducing the users' burden in managing multiple identities and credentials. Typical and generic cookie-based SSO authentication for public WLANs is shown in Fig. 2. The user first shows its identity and credential to service provider A's web server. Then A's web server asks the identity provider's authentication server about the validity of the supplied identity and credential. If the authentication is successful, the result is returned to the user with a cookie, where it is stored in the user terminal. When the user roams into the network of service provider B, B's web server simply retrieves the user's credential from the cookie. Since the web cookie is normally only valid at the web server from which it is issued, SSO systems use special kinds that are valid across multiple domains (so-called common-domain cookies), or they can issue multiple cookies at a time.

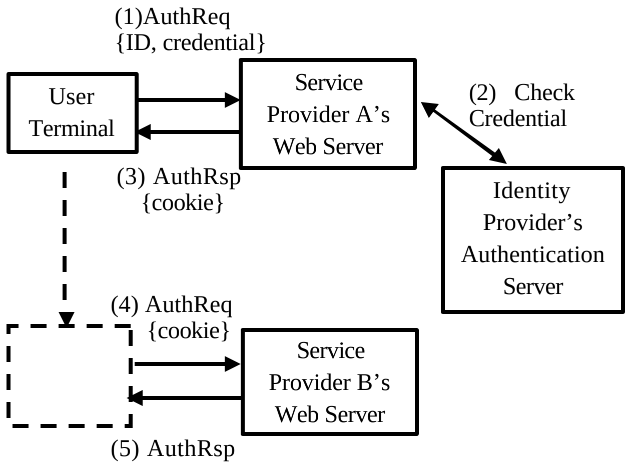

Fig. 2: Single Sign-On in Public WLAN Roaming

We have considered two different authentication models in our architecture: RADIUS and Liberty Architecture. As mentioned in Section 1, the architectural approach is not limited to these two underlying authentication methods, and could easily be extended to add more as necessary. RADIUS utilizes a proxy based authentication scheme; the user sends the authentication data to the service provider and the service provider forwards this to the user's identity provider. Fig. 3 shows a simplified message sequence in RADIUS-based authentication. While RADIUS is widely deployed and provides backward compatibility, the user must expose its identity and credentials to the untrusted service provider, due to the HTTPS-RADIUS conversion process at service provider's web server.

On the other hand, the Liberty Browser Artifact Profile makes use of a redirect-based authentication model (Fig. 4). The user informs the service provider of the name of his identity provider, and then the service provider redirects the user to that identity provider. The user then sends the login information directly to the identity provider, receives its result, and forwards it to the service provider. The identity provider's result includes a pointer to an authentication assertion. Following the pointer, and making use of the SAML protocol [15], the service provider contacts the identity provider to obtain a secure confirmation about the authentication result. Although Liberty-based authentication is more complex than the RADIUS method, it has the advantage of hiding a user's identity and credential from untrusted service providers. It should be noted that to make possible the Liberty flow, a hole must be opened in the service provider's firewall during the authentication to allow the direct communication between the user and the identity provider across the service provider's network.

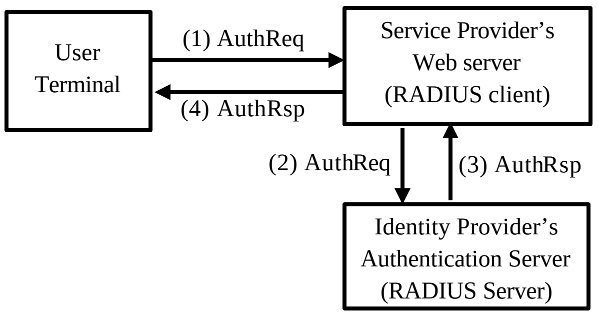
User Terminal
(1) AuthReq
(4) AuthRsp
Service Provider's Web server (RADIUS client)
(2) AuthReq
(3) AuthRsp
Identity Provider's Authentication Server (RADIUS Server)

Fig. 3: Proxy-based RADIUS Authentication

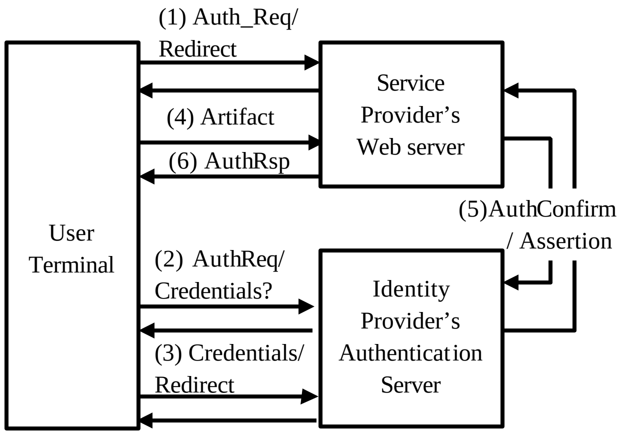
(1) Auth_Req/ Redirect
(4) Artifact
(6) AuthRsp
Service Provider's Web server
(5)AuthConfirm / Assertion
User Terminal
(2) AuthReq/ Credentials?
(3) Credentials/ Redirect
Identity Provider's Authentication Server

Fig. 4: Redirect-based Liberty Browser Artifact Profile Authentication

4. AUTHENTICATION FLOW ADAPTATION

In this section we describe our authentication flow adaptation scheme that selects the appropriate SSO scheme while protecting the privacy of user information in public WLAN environments. In the first subsection, we list the desirable features and give a typical usage scenario. In the following subsections, we describe the key functional components in our framework.

4.1 Desirable Features and a Usage Scenario

Desirable Features

In federated public WLANs, a user roams across networks operated by different providers, each with different billing options and likely requiring different authentication information. Because the user may roam whether they intend to or not, to maximize the security of their user data, they should be notified when roaming is to occur and be forced to input authentication information or manually acknowledge it. The usability of this scheme worsens as the frequency of inter-system roaming increases. To achieve seamless network access, we require the following desirable features for the user interface software:

- Protection of user authentication information against exposure to entities not allowed to see it.
- Minimize user intervention for the sign-on process.
- Support alternative authentication flows and authentication information as required by the underlying service providers.

The first two features require automated access control for the user's authentication information. The access control should be performed according to user-defined policy rules. The policy rules specify entities to be authorized to access each authentication information element using their identifier or attributes (roles).

Policy Rule

```
<policy>
  <authn_info href="…"/>
  <rule>
    <subject>
      <id> … </id>
      <attribute> … </attribute>
    </subject>
    <provisional_action name="…">
    <condition> … </condition>
  <rule>
<policy>
```

Information Element Examples

authn_info	Authentication information file path • authentication method • login ID, password • credit card type, card number, name, expiration date
subject	id: policy identifier attribute: role: ID provider charging option
provisional _action	• user notification • user acknowledgment • user input
condition	• none • through trusted service providers only

Fig. 5: Policy Rule and Information Element Examples

The policy rules may include additional conditions to be satisfied such as the rule is applicable only if the user is connected to the trusted service provider's network, and provisional actions to be fulfilled such as user notification and acknowledgement. Examples of information elements and rules are shown in Fig. 5. In case the user terminal always demands user input or acknowledgment, it is the end user who makes the access control decision according to his security policy and contexts. Our prototype thus far only deploys identifier, attribute (e.g., credentials and credential types), and additional action to be taken (provisional action, e.g., confirmation on the pop-up window) for access control decisions. To achieve the third feature, our scheme enables an authentication server of the WLAN service provider announce its authentication capabilities in response to the authentication request message from the user terminal.

Typical Usage Scenario

To help understand our use of authentication flow adaptation and the role of the policy engine, let's consider an example scenario:

"The user turns on the PDA in his office and automatically gets connected to his department's WLAN. The department WLAN uses 802.1X authentication, and the policy engine permits automatic submission of the pre-shared secret between the user and the department WLAN.

Then the user walks out of the building with his PDA, thus roaming into a campus-wide WLAN network. This network uses a web-based authentication scheme, and is strongly trusted by the user. The policy engine allows the user to automatically submit his identity and credentials to the campus-wide authentication system, thus allowing him to get connected to the campus-wide WLAN network.

As the user leaves the campus and strolls into a cafe, the PDA detects the presence of public WLAN. The policy engine finds both the RADIUS-based and the Liberty-based authentication methods are available on that network, and selects the Liberty-based authentication according to the user's preferences. This service adopts time-based charging, and the user's policy file indicates that automatic roaming should not be performed in such a situation. The policy engine launches a pop-up window on the user terminal and asks the user if he wants to connect to the public WLAN. If the user acknowledges it, the policy engine submits his authentication information to the WLAN service provider's authentication server that provides network access for the cafe, and the user gets access to the Internet. "

4.2 Authentication Sequence Adaptation Overview

In the case that the WLAN service provider supports multiple authentication options, users can select their preferred method. As shown in Fig. 6, in response to an authentication request from a new user that wants to gain access to the external network, the authentication server presents him with the available alternatives and the information required for using each of them. The user then selects the appropriate authentication method based on the policy engine's processing of the user's pre-defined policy (described in the next subsection), and provides the information requested by the service provider for that particular scheme. The service provider's authentication server, switching between alternative methods according to the user preferences and the available information, processes the information. The details of authentication announcement protocol and authentication sequence adaptation will be discussed in a separate paper.

It is worth mentioning that users can still get authenticated regardless of whether the announcement function is installed on the server or whether the user's terminal utilizes the policy engine. Users can get authenticated manually. Thus, legacy user terminals and legacy service provider's authentication servers are readily supported in our architecture.

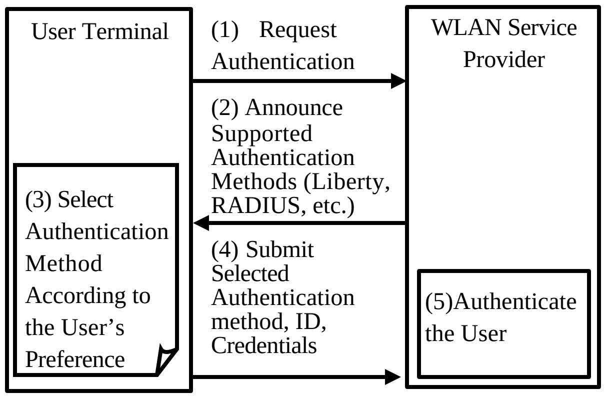
User Terminal
(1) Request Authentication
WLAN Service Provider
(2) Announce Supported Authentication Methods (Liberty, RADIUS, etc.)
(3) Select Authentication Method According to the User's Preference
(4) Submit Selected Authentication method, ID, Credentials
(5)Authenticate the User

Fig. 6: Authentication Sequence Adaptation

4.3 Policy Engine

The policy engine supports policy-based access control for user authentication information. Fig. 7 shows its major components. It is implemented as an independent module which can be invoked through a simple API from a user agent or link layer network access client. An XML-based access control language is used to define access control rules for the user information. Although the policy engine only supports authentication server ID and requested user information as inputs so far, other input parameters, such as service attribute and context, will be supported in future versions.

The access control component performs a policy compliance check based on the input parameters and user-specified access control rules, and outputs granted information along with provisional actions, if any. The provisional action is what must be performed before the corresponding information is sent to the information requestor. In our current implementation, we support only one provisional action, user authorization. If the access control component returns user identity along with the "authorization" provisional action, the policy engine launches a new pop-up window to give the user the option to send the identity to the server. The identity can be sent out only if user pushes the "accept" button. If the access control component returns granted information without provisional action, the information is automatically sent to the server.

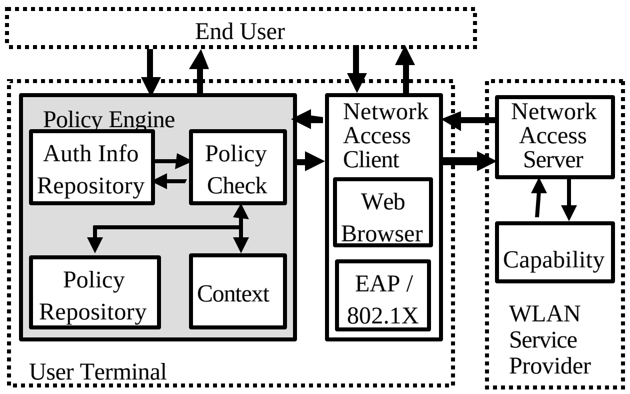

Fig. 7: Policy Engine Block Diagram

The advantages of public WLAN authentication using such a policy engine are:

- Generality: Since the policy engine is built as an independent component with simple APIs, network access clients, like a web browser or an EAP-TLS client, can employ it.
- Scalability: Since users define access control rules using server attributes (e.g., role) and specific server IDs, rule management overhead is kept low even when many WLAN providers with different server IDs are part of the WLAN federation.
- Flexibility: Given a standard vocabulary of the rule specification for user authentication information, users can customize their own access control rules for their authentication information.

Limitation of the current implementation

Currently, the policy engine does not verify authentication information requests to check the validity of input parameters (e.g., server ID). There are several ways to accomplish this. One is to use server's digital signature on the invocation message. Access control reflecting the service attribute or context is not yet supported. When supported, sophisticated sign-on control will be achieved.

5. SECURING WEB-BASED AUTHENTICATION AND ACCESS CONTROL

5.1 Security Threats in Web-based Authentication

Most public wireless LAN systems use web-based authentication schemes, and users can get IP-level network access before showing their identity and credentials. Although this open-style of network authentication enables fine-grained service authorization and accounting options, lack of lower-layer cryptographic bindings yields security vulnerabilities. Examples include:

- Theft of service by spoofing IP or MAC address;
- Eavesdropping because of no data encryption;
- Message alteration because of no message integrity check; and
- Denial of service attack by placing rogue access points.

The key to avoiding those security threats is to have a cryptographic binding between the user and the network. As explained in Section 2, IEEE 802.1X port-based network access control is being deployed in corporate wireless LANs, and it uses such a cryptographic method for user authentication and network access control. Normally IEEE 802.1X adopts conventional closed-style mutual authentication and assumes a pre-shared secret between users and the network. However, we can't assume a pre-shared secret in public wireless LANs to accommodate one-time users that use credit-card authorization, or to provide free contents for non-subscribers. Table 1 summarizes the characteristics of web-based and layer 2-based AAA schemes. Shaded boxes in Table 1 represent the advantage of each authentication scheme.

Table 1: Comparison of Web-based and layer 2-based AAA schemes

	Web-based	**IEEE 802.1X/11i**
Support	Most public wireless LAN service providers	Corporate networks (only in 802 LANs)
Pre-shared Secret	Not necessary (users can use credit-card authorization)	Necessary
Encryption	N/A	Per-station RC4, AES (802.11i)
Authentication	SSL-protected password	EAP-TLS (certificate-based)
Access Control	IP/MAC address	Cryptographic Method
Accounting	Fine-grained	Only at boot time and periodic re-authentication

5.2 Compound Layer 2 and Web Authentication

To ensure cryptographically protected access in public wireless LANs, we have developed a compound layer-2 and Web authentication approach. To use this scheme, the WLAN service provider must have 802.1X-capable access points and authentication servers. The user terminal must also have an 802.1X client, but this stipulation turns out not to be an issue due to the fact that some operation systems bundle an 802.1X client. If this is not the case, free 802.1X clients are available for download [16]. The network may accommodate 802.1X-incapable legacy user terminals to account for backward compatibility. However, allowing 802.1X-incapable clients, thus bypassing link layer authentication, the network becomes vulnerable to common web-based authentication security holes.

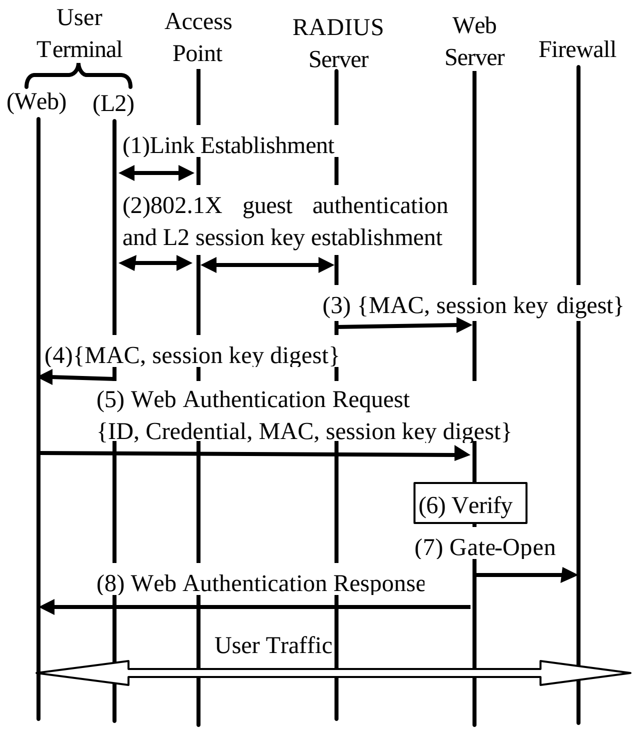
User Terminal
Access Point
RADIUS Server
Web Server
Firewall
(Web)
(L2)
(1)Link Establishment
(2)802.1X guest authentication and L2 session key establishment
(3) {MAC, session key digest}
(4){MAC, session key digest}
(5) Web Authentication Request {ID, Credential, MAC, session key digest}
(6) Verify
(7) Gate-Open
(8) Web Authentication Response
User Traffic

Fig. 8: Compound L2 and Web Authentication Message Sequence

The compound authentication message sequence diagram is shown in Fig. 8. In our scheme, the user terminal first associates with an access point (Step 1) and then establishes a L2 session key using guest (or anonymous) account in EAP-TLS message exchange in IEEE 802.1X authentication (Step 2). A guest account is used in L2 authentication for the reason that we can't assume a pre-shared secret between users and the network in public wireless LAN. It should be noted that the EAP-TLS specification does not mandate client authentication. After Step 2, the encryption key derived from the L2 session key encrypts packets transmitted over the air. The RADIUS server notifies the {MAC address, L2 session key digest} pair to the web server for later use (Step 3). Similarly, the L2 client in the user terminal passes {MAC address, L2 session key digest} to the Web client (Step 4). The inter-process communication in Step 4 can be accomplished by using a customized web client or the policy engine described in the previous section. Then the user terminal sends a Web authentication message to the Web server, with {ID, credential, MAC address, L2 session key digest} quadruplets (Step 5). In Step 5, MAC address and L2 session key digest embedding is required to avoid theft of service by race timing attack from malicious clients. Finally, the Web server verifies the quadruplets (Step 6), changes firewall rules so that the user can access to the external network if the verification succeeds (Step 7), and returns an authentication response to the user terminal (Step 8).

5.3 System Security Analysis

In this section, we give a formal analysis how the proposed compound layer 2 and web authentication scheme can deal with various common security threats.

Theft of Service

A malicious user may spoof the IP and/or MAC address of legitimate user to take over the WLAN service. Address spoofing does not work in our scheme, because each user has its own encryption key established through the 802.1X guest authentication process. The access point simply discards the malicious user's packets if it can't decrypt them. Because all layer 2 data frames are encrypted by per-user key, it is difficult for a malicious user to guess the legitimate user's IP address.

A malicious user can attempt a race timing attack by taking advantage of guest authentication properties. In this case, the malicious user sends a layer 2 authentication request just before a legitimate user's web authentication. The web server can distinguish the malicious user from the legitimate user by checking the {MAC, session key digest} pair embedded in the web authentication request with the one informed by the RADIUS server. Thus the race timing attack can be prevented.

A more sophisticated attack involves a rogue access point between the legitimate user and the legitimate access point. This rogue access point acts as a transparent SSL proxy to fake a legitimate user's network login process. Again, the web server can detect the attack by checking the {MAC, session key digest} pair embedded in the web authentication request message. It can't alter the MAC address or session key digest in the web authentication request message because the message is SSL-encrypted. It should be noted that 802.11i also has a countermeasure for such man-in-the-middle attack by conducting a 4-way handshake immediately after the 802.1X authentication.

Eavesdropping/Message Alteration

It is obvious that eavesdropping and message alteration can be prevented in our scheme, because all L2 data frames are encrypted by per-user key and have message integrity check codes. The per-user key is derived from the L2 session key that is established through EAP-TLS process in 802.1X authentication. Therefore, cracking a legitimate user's per-user key is as difficult as cracking TLS. It is also well known that current 802.11 WEP (wired equivalent privacy) has security vulnerabilities of eavesdropping and message alteration [17]. Countermeasures for such security vulnerabilities are already taken into account in the 802.11i draft under standardization [8]. In the meantime, one can reduce the risk of such vulnerabilities by shortening 802.1X re-keying period.

Denial-of-Service

There are several DoS possibilities in 802.11 wireless LAN [18], such as deauthenticating/disassociating a legitimate user, spoofing power save mode, and faking the carrier sense mechanism. Unfortunately, these DoS possibilities are inherent in the original 802.11 MAC protocol and our scheme does not solve such DoS attacks. Moreover, because every user is given link layer access in our scheme, a malicious user can disturb legitimate user's communication by spoofing the latter's MAC address or flooding frames in layer 2 networks. However, we consider theft of service to be a much more serious problem than DoS attacks, and thus have limited our scope to allow certain DoS scenarios. It is still possible for legitimate users and access points to detect DoS attack and notify it to the network management server.

6. EVALUATION

We have developed a prototype to prove the viability of the architectural concepts we have described above to evaluate the performance of a system that integrates them. In this section we describe this prototype and present its performance.

6.1 Testbed

Our testbed consists of five authentication servers, a WLAN access point, and two user terminals. Two of the authentication servers act as service providers and other two act as identity providers, and the last acts as an 802.1X authentication server. To avoid wireless-specific delay variance, link layer authentication delay and other delay were measured separately. Each server was connected via 100Mb/s Fast Ethernet and link delay was minimal. All of them are implemented on standard Linux or Windows PCs using open-source software.

The hardware specifications of components in the testbed are listed below.

- Identity Provider #1: Pentium III 864 MHz (Linux)
- Identity Provider #2: Pentium II 266 MHz (Linux)
- Service Provider #1: Pentium III 864 MHz (Linux)
- Service Provider #2: Pentium II 266 MHz (Linux)
- User Terminal #1: Pentium IV 1.6 GHz (Windows XP)
- User Terminal #2: Pentium III 1.1GHz (Linux)
- 802.1X Authentication Server: Pentium III 1.1GHz (Linux)
- WLAN Access Point: Cisco AIR-AP352

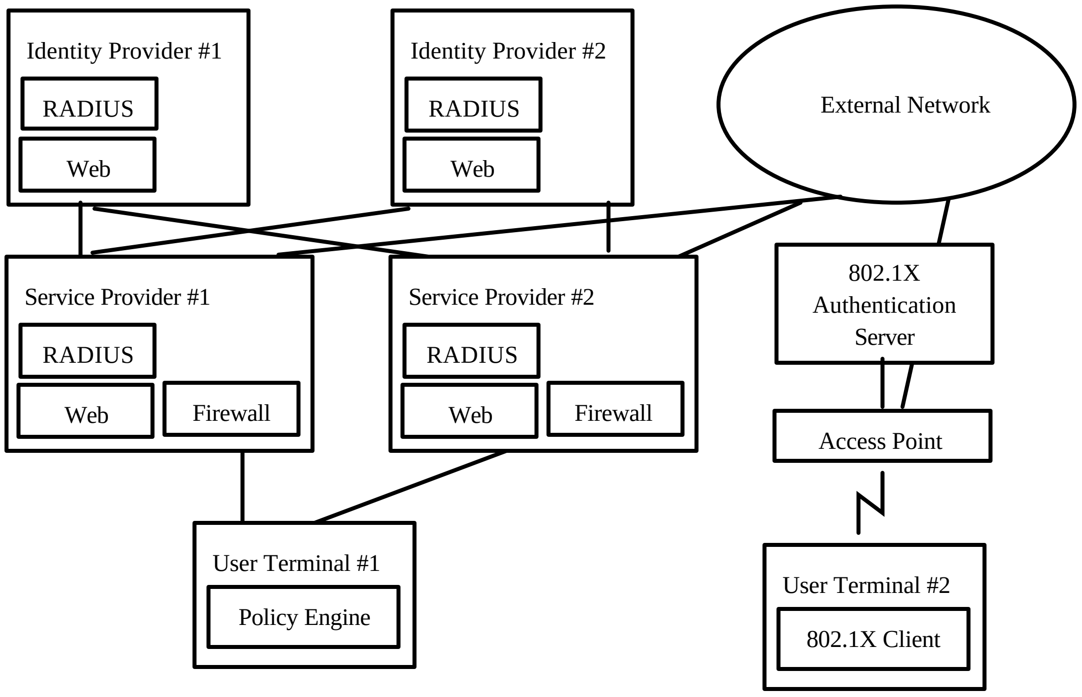

Fig. 9: Testbed Schematics

The open-source software components used in the testbed are as follows.

- GNU RADIUS v0.96.4 (ID/password authentication server)
- FreeRADIUS v0.8.1 (802.1X authentication server)
- XSupplicant v0.6 (802.1X client)
- ForgeNet RADIUS Client 0.9c
- Sun Interoperability Liberty prototype v0.1
- OpenLDAP LDAP Server v2.1.12
- iptables v1.2.6a (Firewall)
- libwww-perl 5.64 (Web client)

6.2 Authentication Latency

To analyze the performance of the different alternatives, we measured the authentication delay for each of the components participating in the authentication process and for the two authentication schemes we considered. In addition, we evaluate two different authentication scenarios: local and remote. In the former, the service provider also plays the role of identity provider, directly authenticating the user. In the latter, a third party plays this role. No transmission delays have been included, as they vary with the distance between entities and are not directly attributable to our implementation.

The following table shows the results obtained for roaming authentication using the policy engine to submit the authentication data. The total delay was less than 2 seconds in the worst case, in our view an acceptable authentication latency for WLAN users.

The delay can be divided in four components: link layer authentication, firewall redirection, policy engine and web authentication. The link layer authentication delay is due to the establishment of a secure layer 2 communication using the IEEE 802.1X protocol. Firewall redirection delay includes the detection of an unauthenticated user and his redirection to the web authentication interface using SSL. The Policy Engine delay comprises the selection in the client side of the authentication method to use and the information to submit depending on the user defined policies and the communication context. Finally, the web authentication delay is the elapsed time from when the user sends his authentication data to the service provider to when the authentication is completed and the user is notified.

The most significant delay in the web authentication segment is the Liberty remote case. It is due to the exchange of SAML messages between service provider and identity provider to confirm the authentication of the user, which have to be signed and verified using public-private key cryptography. This is not needed when the authentication is local, since the same entity plays the role of service provider and identity provider.

The next most significant delay was caused by the policy engine processing the authentication rules in the client. It is due to the parsing of XML files with the information, which is done every time the engine is invoked. Alternative methods of implementation could be used to reduce this delay.

The link layer (802.1X) authentication delay was as small as 0.12 sec, and the additional overhead of verifying {MAC, L2 key digest} at web authentication was smaller than 1 msec.

Table 2: Authentication Delay Profile

	Proxy-based (RADIUS)		Redirect-based (Liberty)	
	Local	**Remote**	**Local**	**Remote**
Web Authentication	0.098	0.102	0.089	1.381
Policy Engine	0.318			
Firewall Redirection	0.086			
Link Layer (802.1X) Authentication	0.124			
Total	0.626	0.630	0.617	1.909

7. CONCLUSION

Dynamic selection of the authentication method and the identity provider will play a key role in confederating public wireless LAN service providers under different trust levels and with alternative authentication schemes. The proposed authentication adaptation framework makes it possible to accommodate multiple authentication methods. We exploited two different industry-standard single sign-on authentication schemes in public wireless LANs: RADIUS and Liberty Architecture. A client-side policy engine enables the user to select which of the alternate single sign-on authentication schemes to use. The policy engine also protects the user's privacy information by forcing him to input authentication information manually when he roams into a service provider he weakly trusts. In addition, we developed a compound layer 2 and Web authentication scheme to prevent theft of service, eavesdropping, and message alteration in public wireless LANs. To demonstrate the feasibility of our approach, we developed a single sign-on prototype system. The measured authentication delay values ranged from 0.6 to 1.9 sec depending on the authentication types, and they were all small enough for practical use.

8. ACKNOWLEDGMENTS

The authors would like to thank David Wagner for valuable suggestions on trust relationship, and Manish Shah for carefully reviewing the paper. Yasuhiko Matsunaga is supported by NEC Corporation, as a visiting industrial fellow at University of California, Berkeley. Takashi Suzuki was supported by NTT DoCoMo, as a visiting industrial fellow at University of California, Berkeley, until May 2003. Funding for this work was provided in part by the California MICRO Program, with matching industrial support from Sprint and Ericsson.

9. REFERENCES

[1] HotSpotList.com, http://www.hotspotlist.com/

[2] IETF, RFC 2865 "Remote Authentication Dial In User Service (RADIUS)", June 2000.

[3] Liberty Alliance Project, "Liberty Architecture Overview", version 1.1, January 2003.

[4] Wi-Fi Alliance, "Best Current Practices for Wireless Internet Service Provider (WISP) Roaming", ver. 1.0, 2003.

[5] S. Hada and M. Kudo, "Access Control Model with Provisional Actions", IEICE Trans. Fundamentals, Vol. E84-A, No.1, Jan. 2001.

[6] OASIS eXtensible Access Control Markup Language, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.

[7] IEEE Std 802.1X-2001, "Port-Based Network Access Control", June 2001.

[8] IEEE Std 802.11i/D4.0, "Medium Access Control (MAC) Security Enhancements", May 2003.

[9] IETF, RFC 2716, "PPP EAP TLS Authentication Protocol", Oct. 1999.

[10] Internet-Draft, "EAP Tunneled TLS Authentication Protocol", draft-ietf-pppext-eap-ttls-02.txt, work in progress.

[11] IETF RFC 2402, "IP Authentication Header", Nov. 1998.

[12] D. Jablon, "Strong Password-Only Authenticated Key Exchange", Computer Communication Review, Vol.26, 1996.

[13] http://srp.stanford.edu/

[14] V. Bahl, A. Balachandran, S. Venkatachary, "The CHOICE Network: Broadband Wireless Internet Access In Public Places", Microsoft Technical Report, MSR-TR-2000-21, Feb. 2000.

[15] OASIS, "Assertions and Protocol for the OASIS Assertion Markup Language (SAML)", Committee Specification 01, May 2002.

[16] http://www.open1x.org/

[17] N. C-Winget, R. Housley, D. Wagner, J. Walker, "Security flaws in 802.11 data link protocols", Communications of the ACM, 46(5), May 2003, pp. 35-39

[18] J. Bellardo and S. Savage, "802.11 Denial-of-Service Attacks: Real Vulnerabilities and Practical Solutions", to appear in Proceedings of the USENIX Security Symposium, August 2003.

[19] IETF, RFC2759 "Microsoft PPP CHAP Extensions, Version 2", Jan. 2000.